1B5-03

災害救助における燃焼家屋クラスタリングの一考察

Consideration of Clustering burned building for RoboCupRescue

本地 健一 Ken'ichi Honji　伊藤 暢浩 Nobuhiro Ito　犬塚 信博 Nobuhiro Inuzuka　和田 幸一 Koichi Wada

名古屋工業大学 電気情報工学科
Electrical and Computer Engineering, Nagoya Institute of Technology

We implimented the idea that some adjacent burning buildings are recognized a cluster. But, the clustering, to classify each elements into some clusters, needs huge quantities of time. On the other hand, agents must decide action in short time. In this paper, We propose quick clustering with the learning. We show that this technique can classify burning buildings as clusters in short time.

1. はじめに

近年，大地震などの大規模災害に対する関心が高まり，その取り組みの一つである RoboCupRescue プロジェクトに注目が集まっている．我々はこのプロジェクトのシミュレーションプロジェクトにおける災害救助エージェントを対象として研究をおこなっている [1].

本研究では，特にその中で火災エージェントの設計時に必要となる燃焼家屋のクラスタリングの問題を取り扱う．エージェントが燃焼家屋を消火するとき，各家屋に対して消火するよりも，クラスタとして分類し消火する方が効率が良いと考えられる．しかしこの際，従来のクラスタリング手法 [2][3] では，火災のように対象となる要素点が追加されたり消滅したりすることを想定していないため，毎回クラスタリングが必要で，一回毎に要素点数 n に対して $O(n^3)$ もの計算時間を必要とする．これを現在の Rescue シミュレーション上で実装すると，クラスタリングに時間がかかりすぎる．

そこで，我々は学習を用いたクラスタリング手法を提案し，クラスタリングの高速化をおこなう．ただし，本手法は要素が変化した場合は従来と同様に再クラスタリングをおこない，代わりに対象とする要素数を減らすことにより高速化を実現するものである．提案手法は学習段階と，分類段階に分かれており，学習段階では，手本となる従来手法で作られたクラスタの最外郭の点（輪郭点と呼ぶ）を周辺パターンから発見する学習をおこなう．分類段階では，学習で得た輪郭点の集合のみをクラスタリングして，より高速にクラスタ分類をする．

また実験では，実際に従来手法よりも高速にクラスタリングができているかどうかを示し，高速化したことによってクラスタ分類に誤りがないかどうかも検証する．

2. 既存のクラスタリング手法とその問題点

クラスタリングとは，ある条件を満たした要素同士が集まってできるクラスタと呼ばれる集合に，各要素を分類することである．クラスタリング手法は階層的クラスタリングと非階層的クラスタリングに大きく分かれ，その中で主に図 1 に示すような各手法が存在する [4].

本研究では二次元平面上の点集合に対して用いるため，Agglomerative が最も適している．Agglomerative クラスタリングは，ボトムアップ型のクラスタリング手法である (表 1).

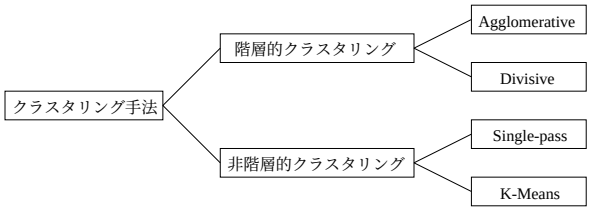

図 1: クラスタリングの分類

既存のクラスタリングはクラスタを生成するのに要素数 n に対して，$O(n^3)$ の計算時間を必要とする．

3. 学習を用いたクラスタリング手法

既存のクラスタリング手法の問題点を改善するためには，従来手法を次のような点で改善する必要がある．

- 要素が変化しても一から再クラスタリングをおこなわず，変化分だけクラスタリングする
- クラスタリングする要素数をある程度の数に減らすことができるようにする

本研究では，学習を用いて，クラスタリングする要素数を減らし，その高速化をおこなう手法を採用する．また学習によってクラスタリングする要素点は，クラスタの輪郭を形成するような点のみとした．これは全ての要素点に比べて非常に少ない要素点で構成されているため，そのクラスタリングは従来よりもはるかに高速におこなえる．この段階で各クラスタは輪郭のみのリング状を形成しているので，残りの要素はリングが二重以上の輪になっていない限り，必ずそのリングの内側にある．そのため残りの要素をクラスタ分類することは容易である．

つまり，この手法は従来に比べて非常に高速にクラスタリングができると考えられる．具体的には次に述べる**学習段階**, **分類段階**の二つの段階によるクラスタリングをおこなう．

3.1 学習段階

この段階は，実際にクラスタリングをおこなう分類段階の前処理となる．従来手法で生成されたクラスタを手本にして，学習により次の 3 ステップで点集合からクラスタの輪郭，つまりクラスタの外側の点 (輪郭点) を抽出できるようにする．

連絡先: 本地 健一，名古屋工業大学，愛知県名古屋市昭和区御器所町，TEL:052-735-5408，FAX:052-735-5408，E-mail:honji@phaser.elcom.nitech.ac.jp

表 1: 階層型 Agglomerative クラスタリングのアルゴリズム

⟨**Agglomerative Hierarchical Clustering** $AHC(B)$ ⟩
入力：n 個の要素点 $(b_1, b_2, \cdots, b_n)$ の集合 B
出力：クラスタ集合 $\mathcal{C} = \{C_1, C_2, \cdots, C_m\}$($m$:クラスタ数)
対象とする n 個の要素点 $(b_1, b_2, \cdots, b_n)$ 全てをそれぞれを要素数 1 のクラスタ $C_i (i = 1, 2, \cdots, n)$ とし，
$\mathcal{C} = \{C_1, C_2, \cdots, C_n\}$ とする
距離 $D_{ij}(C_i, C_j)$：直交座標系での二つのクラスタ C_i, C_j 間の距離
D'：クラスタリングにおける閾値
D_{min}：距離 D の最小値
C_{min_1}, C_{min_2}：距離 D が最小となる時のクラスタの組

$D_{min} = \infty$
for i=1 to n-1
  for j=i+1 to n
    if $D_{ij}(C_i, C_j) < D_{min}$ then
      D_{min}:=$D_{ij}(C_i, C_j)$
      C_{min_1}:=C_i
      C_{min_2}:=C_j
while $D_{min} <= D'$ do begin
  $\mathcal{C}$:=$\mathcal{C} - \{C_{min_1}, C_{min_2}\}$
  $\mathcal{C}$:=$\mathcal{C} \cup (C_{min_1} \cup C_{min_2})$ /* 結合 */
  /* 結合により $|\mathcal{C}|$ は 1 減少 */
  $D_{min} = \infty$
  for i=1 to $|\mathcal{C}|$-1
    for j=i+1 to $|\mathcal{C}|$
      if $D_{ij}(C_i, C_j) < D_{min}$ then
        D_{min}:=$D_{ij}(C_i, C_j)$
        C_{min_1}:=C_i
        C_{min_2}:=C_j
end
return $\mathcal{C}$

step1. 状態のパターン分類

学習対象となる要素点の半径 r 以内の領域を $\mathcal{A}$ とし，これを適当な方位数 d で領域 $A_1, A_2, \cdots, A_d$ に分割する．そして，方位ごとの領域 A_i における要素点の密集度 p_{A_i}(要素領域に対する存在する要素の数の割合) を適当な分類により p 段階に分ける．この分類した周囲の情報 p^d パターンを状態数とする．(表 2, 図 2)

表 2: 状態分類のアルゴリズム

⟨ **状態のパターン分類** $getS(b)$ ⟩
入力：対象となる要素 b
出力：状態 s
d：分割方位数
$\mathcal{A} = \{A_1, A_2, \cdots, A_d\}$
$A_i = \{b_1, b_2, \cdots, b_{l_i}\}$
l_i：各領域 A_i 内に存在する要素数
p_{A_i}：領域 A_i における要素の密度 (1〜p の整数値)

/* p^d パターンの状態表現 */
$s := \sum_{i=1}^{d} p_{A_i} * p^{d-i}$
return s

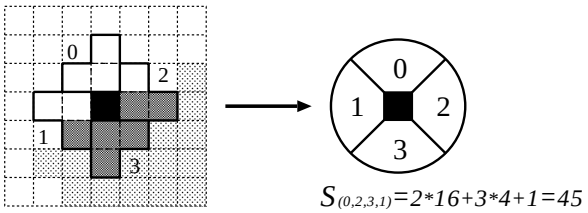

図 2: 状態の決定 (r=2,d=4,p=4 のとき)

step2. 行動選択

行動選択は，step1. で求めた状態を持つ点が輪郭点であるかどうかの判断でおこなわれるとする．(表 3)

表 3: 行動選択のアルゴリズム

⟨ **行動選択** $decideA(s)$ ⟩
入力：対象となる要素 b の状態 s
出力：輪郭かどうか $(True, False)$
確率 $P(s)$：状態 s で輪郭点である確率
乱数 rand：0 から 1 の間のランダムな実数

if rand< $P(s)$ then
  return $True$ /* 確率的選択 */
else
  return $False$

step3. 行動に対する報酬

step2. での行動 (輪郭かどうかの判断) が正しかった場合に報酬を与える．(表 4)

以上の step により学習を前処理としておこなっておくことで，次回からは与えられた集合 $B = \{b_1, b_2, \cdots, b_n\}$ に含まれ

表 4: 報酬獲得のアルゴリズム

〈 **報酬の獲得** $update(b, s, a, P(s))$ 〉
入力：対象となる要素 b の状態 s, 行動結果 a
出力：更新後の確率 $P'(s)$
報酬：P_r

if b が輪郭点である then
  $a' = True$
else
  $a' = False$
if $a = a'$ then
  if $a = True$ then
    $P'(s) := P(s) + P_r$
  else
    $P'(s) := P(s) - P_r$
return $P'(s)$

る要素点 b_i をそれぞれの状態 s_i にパターン分類でき，学習結果からそれが輪郭点かどうかを判定し，k 個の輪郭点の集合 $B' = \{b_1, b_2, \cdots, b_k\}$ を取り出すことができるようになる．この学習段階は，クラスタリングの準備段階であるので，学習にかかる時間はクラスタリングの時間に含まない．

3.2 分類段階

この段階が本手法の核となる部分であり，学習段階で得られた知識を用いて次の 3 ステップでクラスタリングをおこなう．またそのアルゴリズムは表 5 のようになる．

step1. 学習結果を用いたクラスタの輪郭点の同定
輪郭同定の学習結果を用いて，与えられた要素集合からクラスタの最外郭の点 (輪郭点と呼ぶ) を見つける (図 3).

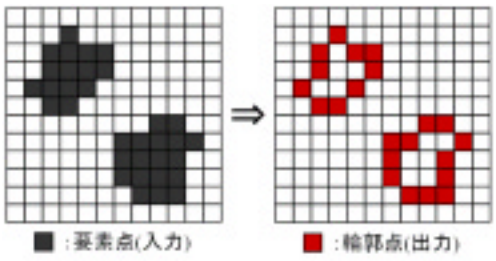

図 3: Step1 の流れ

step2. 得られた輪郭点集合に対するクラスタリング
step1. により得られた輪郭点集合に対して既存のクラスタリング手法によりクラスタリングをおこなう．ここで，輪郭点はリング状のクラスタを形成する (図 4).

step3. 輪郭点でない残りの要素のクラスタ分類
step2. により得られたクラスタに残りの要素を分類する．分類方法は，残った任意の要素をそこから最も近い輪郭点に属するクラスタに割り当てる (図 5).

以上の step により生成されたクラスタはその計算時間が次のようになる．まず各ステップでは，輪郭の取り出しに $O(n)$, 取り出した輪郭点集合をクラスタリングするのに $O(k^3)$, 残っ

表 5: 分類段階のアルゴリズム

〈 **要素点集合からのクラスタリングアルゴリズム** 〉
入力：$B = \{b_1, b_2, \cdots, b_n\}$(与えられる要素点 b の集合)
出力：$\mathcal{C}$(クラスタの集合)
B'：輪郭点である要素点 b の集合 (初期状態は空集合)

/* **step1** 輪郭点の同定 */
for i=1 to n begin
  $s := getS(b_i)$ (状態のパターン分類)
  $a := decideA(s)$ (輪郭点かどうかの決定)
  if $a = True$ then 集合 B' に b_i を追加
end
/* **step2** 輪郭点集合のクラスタリング */
/* $B' = \{b_1, b_2, \cdots, b_k\}$($k$:輪郭点数) である */
$\mathcal{C} := AHC(B')$
/* $\mathcal{C} = \{C_1, C_2, \cdots, C_m\}$($m$:クラスタ数) である) */
/* **step3** 輪郭点でない残りの要素のクラスタ分類 */
/* 輪郭点でない，残る n-k 個の要素点 $(b_1, b_2, \cdots, b_{n-k})$ 全てをそれぞれを 1 つのクラスタ $C'_i (i = 1, 2, \cdots, n\text{-}k)$ とし，$\mathcal{C}' = \{C_1, C_2, \cdots, C_{n-k}\}$ とする */
for i=1 to n-k begin
  $D_{min} = \infty$
  for j=1 to m
    if $D_{ij}(C_i, C_j) < D_{min}$ then
      D_{min}:=$D_{ij}(C_i, C_j)$
      C_{min_1}:=C_i
      C_{min_2}:=C_j
  $\mathcal{C}'$:=$\mathcal{C}' - C_{min_1}$
  $\mathcal{C}$:=$\mathcal{C} - C_{min_2}$
  $\mathcal{C}$:=$\mathcal{C} \cup (C_{min_1} \cup C_{min_2})$ /* 結合 */
  /* 結合により $|\mathcal{C}'|$ は 1 減少 */
end

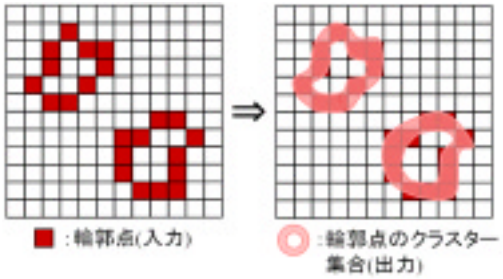

図 4: Step2 の流れ

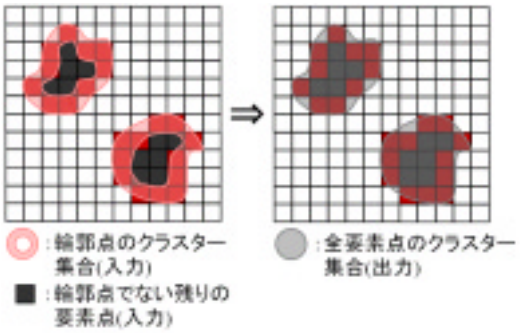

図 5: Step3 の流れ

た要素の分類に $O(k*(n-k))$ の計算時間を必要とする．したがって，全体の計算時間は $O(k^3)$ となる．ここで輪郭点の個数 k は，最悪の場合は n と等しくなるので計算時間を短縮できない．しかし輪郭が最も短くなる場合を考えると，そのときはクラスタが一つで，その形状が円となる．このとき n は円の面積に，輪郭点の個数 k は円周に相当することから $k=2\sqrt{\pi n}$ となり，クラスタリングにかかる計算時間を最適な $O(n\sqrt{n})$ にまで短縮することができる．

4. 実験と結果

提案手法が高速，かつ正確にクラスタができているかどうかを検証するために，既存のクラスタリング手法と提案したクラスタリング手法でそれぞれ生成されたクラスタから，次の二つの点で比較をおこなった．

- 各クラスタリングにかかる時間を，横軸を要素数，縦軸を時間にとって比較した
- 各手法で分類されたクラスタで，輪郭かどうかの判断の一致率と，クラスタ面積の一致率を比較した

環境として，50 × 50 の格子状空間を考え，それぞれの格子を要素点とし，時間と共に一定確率で要素が増える様子を再現した．各計算時間は，実際のクラスタリングをおこなう部分のみの時間を計測し，要素の増大と共に増加してゆく時間を，それぞれの手法でグラフ化した (図 6)．

一方，提案手法で生成されたクラスタが正しく分類されているかどうかを検証するために，従来手法で生成されたクラスタとの面積を比較している．輪郭点の学習が正しくおこなわれていれば，その輪郭点からクラスタリングをおこなったとき，できるクラスタの数や形は従来手法で生成されたクラスタと一致するはずである．よって，どのくらい一致するかでクラスタが正しく分類されているかどうかを検証することができる．

図 6 から，既存の手法が $O(n^3)$ かかっているのに対して，提案手法は $O(n\sqrt{n})$ に近い時間で生成できていることが分かる．また表 6 から，p=2,3 のときに高い確率で既存の手法によって生成されるクラスタとほぼ同じ分類をすることができた．

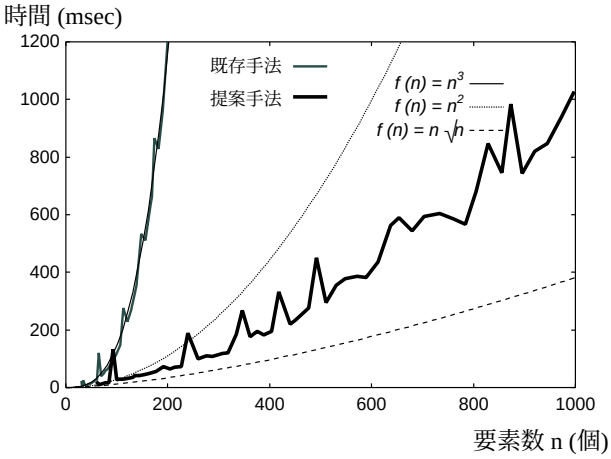

図 6: クラスタ生成にかかる時間

表 6: 生成されたクラスタの一致率

存在率の段階数 p の値	2	3	4
要素点一致率平均 (%)	89.5	89.6	93.6
クラスタ面積一致率平均 (%)	96.7	89.9	66.9

5. まとめと今後の課題

本研究では，学習を用いてクラスタリングをおこなう方法を提案した．まず，周辺の情報をパターンに分類し輪郭かどうかを行動にとることにより，クラスタの輪郭を同定した．次に，学習結果を用いて，全要素点から輪郭点を抽出し，この輪郭点を既存の手法でクラスタリングした．残りの要素は近い輪郭点に割り当てることで全ての要素をクラスタに分類することができた．この手法を用いることで，既存手法より短い計算時間でのクラスタリングが実現できた．また，高い精度で既存手法のクラスタリング結果を再現できた．

今後の課題としては，クラスタの内部に空洞ができてしまうドーナツ型のクラスタに対しても提案手法が使えるようにする必要がある．また，再クラスタリングをおこなわず変化分だけクラスタリングする手法についても考えていくべきである．

参考文献

[1] RoboCupRescue Official Site, http://www.r.cs.kobe-u.ac.jp/robocup-rescue/index.html

[2] 宮本 定明: クラスター分析入門 -ファジィクラスタリングの理論と応用-, 森北出版 (1999).

[3] Michael R. Anderberg: クラスター分析とその応用, 内田老鶴圃 (1988).

[4] BCI Clustering Software – クラスタリングの種類, http://www.adamnet.co.jp/scs/products/bci/overview.html